clp.r

denis

2021-07-12

#!/usr/bin/r  
  
# These properties in fact define a more general inner product for other kinds of  
# mathematical objects for which an addition, an additive identity, and a mulch-  
# application by a scalar are defined. (We should restate here that we assume the  
# vectors have real elements. The dot product of vectors over the complex weld  
# is not an inner product because, if x is complex, we can have x T x = 0 when  
call("output", "x", "t")

## output("x", "t")

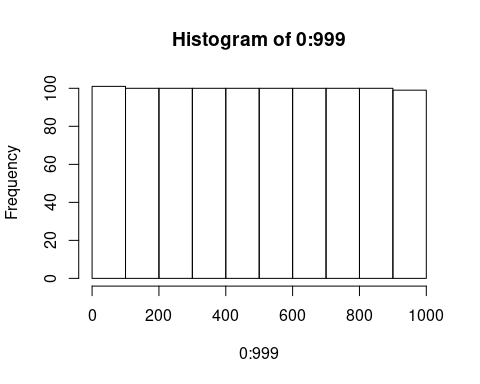
output <- c(10, 20)  
output

## [1] 10 20

# 2 Vectors and Vector Spaces  
# x = 0. An alternative definition of a dot product using complex conjugates is  
# an inner product, however.) Inner products are also defined for matrices, as we  
# will discuss on page 74. We should note in passing that there are two different  
# kinds of multiplication used in property 3. The erst multiplication is scalar  
# multiplication, which we have defined above, and the second multiplication is  
# ordinary multiplication in IR. There are also two different kinds of addition  
# used in property 4. The erst addition is vector addition, defined above, and  
# the second addition is ordinary addition in IR. The dot product can reveal  
# fundamental relationships between the two vectors, as we will see later.  
# A useful property of inner products is the Cauchy-Schwartz inequality:  
IR <- later::global\_loop()  
IR

## <event loop>  
## id: 0

# This relationship is also sometimes called the Cauchy-Bunyakovskii-Schwartz  
# inequality. (Agustin-Louis Cauchy gave the inequality for the kind of dis-  
# create inner products we are considering here, and Victor Bunyakovskii and  
# Herman Schwartz independently extended it to more general inner products,  
# defined on functions, for example.) The inequality is easy to see, by erst ob-  
# serving that for every real number t,  
t <- hist(0:999)



# where the constants a, b, and c correspond to the dot products in the   
# preceding equation. This quadratic in t cannot have two distinct real roots.   
# Hence the discriminant, b 2 − 4ac, must be less than or equal to zero;   
# that is,  
a <- c(10, 5, 20, 6, 35, 8, 45, 9)  
b <- c(10, 5, 20, 6, 35, 8, 45, 9)  
a + sin(a)

## [1] 9.455979 4.041076 20.912945 5.720585 34.571817 8.989358 45.850904  
## [8] 9.412118

b + sin(b)

## [1] 9.455979 4.041076 20.912945 5.720585 34.571817 8.989358 45.850904  
## [8] 9.412118

# By substituting and taking square roots, we get the Cauchy-Schwarz inequal-  
# ity. It is also clear from this proof that equality holds only if x = 0   
# or if y = rx,  
# for some scalar r.  
r <- 0  
r + sin(1)

## [1] 0.841471

# 2.1.5 Norms  
# We consider a set of objects S that has an addition-type operator, +, a cor-  
# responding additive identity, 0, and a scalar multiplication; that is, a   
# multi-  
# plication of the objects by a real (or complex) number. On such a set, a norm  
# is a function, from S to IR that satisﬁes the following three conditions:  
IR = 20  
  
# 1. Nonnegativity and mapping of the identity:  
# if x = 0, then x > 0, and 0 = 0 .  
if (!missing(IR)) {  
 IR <- list("input", "S4")  
 c(IR)  
}

## [[1]]  
## [1] "input"  
##   
## [[2]]  
## [1] "S4"

# 2. Relation of scalar multiplication to real multiplication:  
# ax = |a| x for real a.  
ax <- c(8)  
ax + sin(ax)

## [1] 8.989358

# 3. Triangle inequality:  
# x + y ≤ x + y.  
trigamma(ax)

## [1] 0.133137

# (If property 1 is relaxed to require only x ≥ 0 for x = 0, the function is  
# called a seminorm.) Because a norm is a function whose argument is a vector,  
# we also often use a functional notation such as ρ(x) to represent a norm.  
for (ax in sin(ax)) {  
 c(ax)  
}  
ax

## [1] 0.9893582

# Sets of various types of objects (functions, for example) can have norms,  
# but our interest in the present context is in norms for vectors and (later)  
# for matrices. (The three properties above in fact deﬁne a more general norm  
# for other kinds of mathematical objects for which an addition, an additive  
# identity, and multiplication by a scalar are deﬁned. Norms are deﬁned for  
# matrices, as we will discuss later. Note that there are two diﬀerent kinds of  
# multiplication used in property 2 and two diﬀerent kinds of addition used in  
# property 3.)  
later::with\_loop(loop = 0:999, expr = "compile")

## [1] "compile"

# A vector space together with a norm is called a normed space.  
# For some types of objects, a norm of an object may be called its “length”  
# or its “size”. (Recall the ambiguity of “length” of a vector that we mentioned  
# at the beginning of this chapter.)  
length(ax)

## [1] 1

# L p Norms  
# There are many norms that could be deﬁned for vectors. One type of norm is  
# called an L p norm, often denoted as · p . For p ≥ 1, it is deﬁned as  
Lp <- normalizePath(path = ".", winslash = "\\")  
Lp

## [1] "/home/denis/projects/perl/perlstyle/matrix/bin/desktop"

# This is also sometimes called the Minkowski norm and also the Hölder norm.  
# It is easy to see that the L p norm satisﬁes the ﬁrst two conditions above.   
# For  
# general p ≥ 1 it is somewhat more diﬃcult to prove the triangular inequality  
# (which for the L p norms is also called the Minkowski inequality), but for   
# some  
# special cases it is straightforward, as we will see below.  
p = 16  
p > 1

## [1] TRUE

# x 1 = i |x i |, also called the Manhattan norm because it corresponds  
# to sums of distances along coordinate axes, as one would travel along the  
# rectangular  
# street plan of Manhattan.  
retracemem(p, previous = NULL)  
  
# x 2 =  
# i x i , also called the Euclidean norm, the Euclidean length, or  
# just the length of the vector. The L p norm  
# is the square root of the inner  
# product of the vector with itself: x 2 =  
# x, x  
length(p)

## [1] 1

# x ∞ = max i |x i |, also called the max norm or the Chebyshev norm. The  
# L ∞ norm is deﬁned by taking the limit in an L p norm, and we see that it  
# is indeed max i |x i | by expressing it as  
L <- max(p, na.rm = FALSE)  
L

## [1] 16

# x ∞ = lim x p = lim  
# p→∞  
  
  
# p→∞  
# p 1  
#|x i |  
# p  
#i  
  
# 1  
# x i p p  
# = m lim  
#  
# p→∞  
# m  
# i  
# with m = max i |x i |. Because the quantity of which we are taking the p th  
# root is bounded above by the number of elements in x and below by 1,  
# that factor goes to 1 as p goes to ∞.  
m <- max(p, na.rm = FALSE)  
m + sin(m)

## [1] 15.7121